

# Instituto Tecnológico de Cancún



## **Materia:**

**Fundamentos de Telecomunicaciones**

## **Tarea:**

**Proyecto Sistema de comunicación**

**Alumno:** Aguilar Moreno Jorge Axel

**Docente:** Ismael Jiménez Sánchez

**Horario:** 17:00 – 18:00

**Ing. Sistemas Computacionales**

**5.-Semestre**

## Introducción

En este proyecto trataremos de aplicar un sistema de comunicación entre dos maquinas virtuales atreves de diversos programas para realizar esta presentación

A continuación, mostrare los siguientes programas para poder realizar este Proyecto Sistema de comunicación

Requisitos:

- Scripts en python
- GNS3
- VirtualBox
- Vagrant
- Putty



### Fase1 - Instalar 2 centos7 en VirtualBox usando vagrant.

Abrimos en PowerShell para que podamos trabajar con el **vagrant** para poder descargar las imágenes de CentOS o el sistema operativo CentOS 7 desde vagrant.

También nos da la opción en que maquina se va manejar en nuestro caso elegiremos **VirtualBox** para descargarlo.

```
PS C:\Users\jaamT\jaamT> vagrant box add centos7
==> box: Loading metadata for box 'centos/7'
    box: URL: https://vagrantcloud.com/centos/7
This box can work with multiple providers! The providers that it
can work with are listed below. Please review the list and choose
the provider you will be working with.

1) hyperv
2) libvirt
3) virtualbox
4) vmware_desktop

Enter your choice: 2
==> box: Adding box 'centos/7' (v2004.01) for provider: libvirt
    box: Downloading: https://vagrantcloud.com/centos/boxes/7/versions/2004.01/providers/libvirt.box
Download redirected to host: cloud.centos.org
Progress: 2% (Rate: 582k/s, Estimated time remaining: 0:13:21)
```

Ya después de haber descargado el CentOS 7 lo guardaremos en una carpeta para poder trabajarlo desde la máquina virtual.

```
Prueba la nueva tecnología PowerShell multiplataforma http
PS C:\Users\jaamT> mkdir vagrant-inicio

Directorio: C:\Users\jaamT

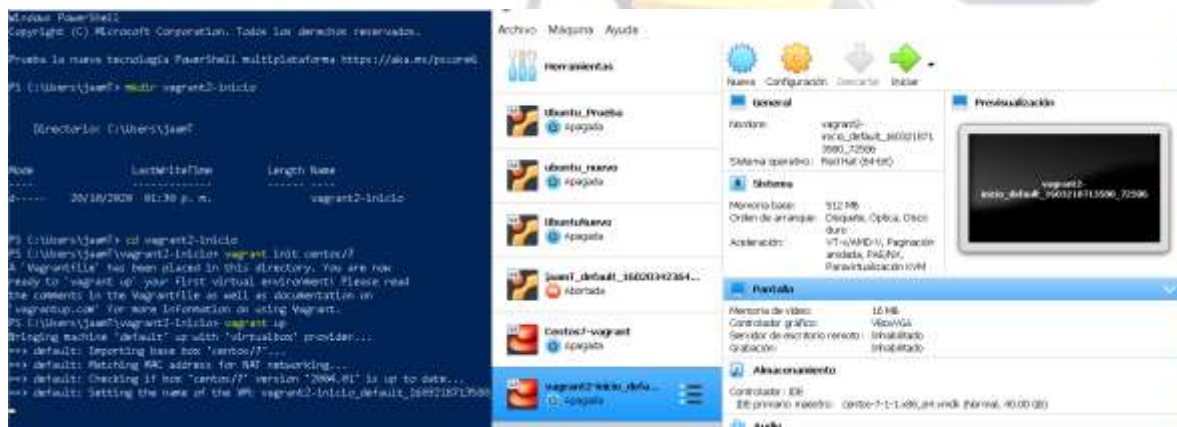
Mode                LastWriteTime         Length Name
----                -
d-----          20/10/2020  12:44 p. m.             vagrant-i

PS C:\Users\jaamT> cd vagrant-inicio
PS C:\Users\jaamT\vagrant-inicio> vagrant init centos/7
A `Vagrantfile` has been placed in this directory. You are
ready to `vagrant up` your first virtual environment! Plea
the comments in the Vagrantfile as well as documentation o
`vagrantup.com` for more information on using Vagrant.
PS C:\Users\jaamT\vagrant-inicio> █
```

```

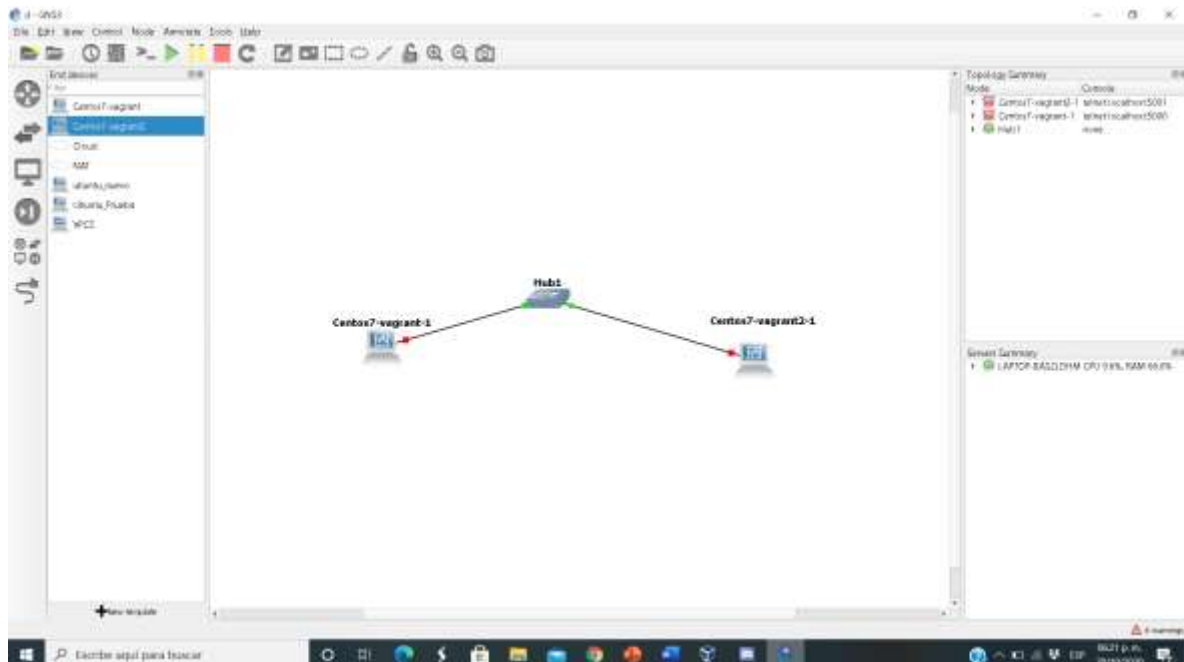
`vagrantup.com` for more information on using Vagrant.
PS C:\Users\jaamT\vagrant-inicio> vagrant up
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Importing base box 'centos/7'...
==> default: Matching MAC address for NAT networking...
==> default: Checking if box 'centos/7' version '2004.01' is up to date...
==> default: Setting the name of the VM: vagrant-inicio_default_1603216035669_89744
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
    default: Adapter 1: nat
==> default: Forwarding ports...
    default: 22 (guest) => 2222 (host) (adapter 1)
==> default: Booting VM...
==> default: Waiting for machine to boot. This may take a few minutes...
    default: SSH address: 127.0.0.1:2222
    default: SSH username: vagrant
    default: SSH auth method: private key

```



En esta parte final podemos ver que vagrant a logrado instalar los CentOS 7 en el **VirtualBox** ahora solo procederemos a apagar las maquinas para seguir con la demostración des este proyecto

## Fase2 - Conectar en GNS3, las dos VMs de CentOS con un switch ethernet



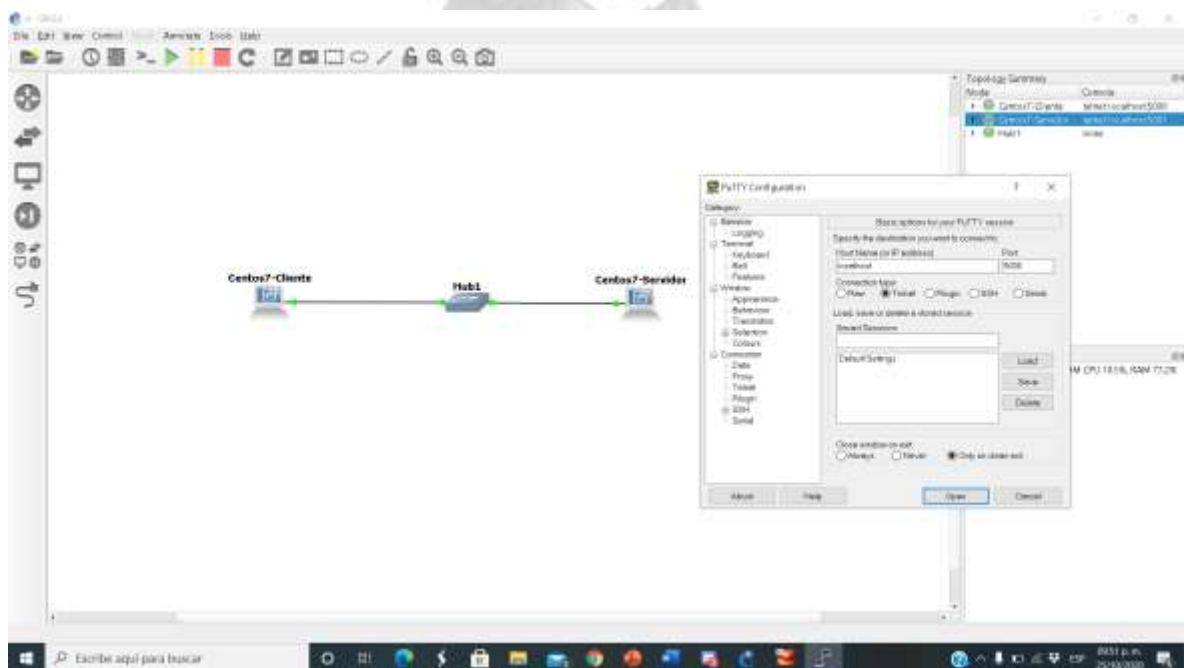
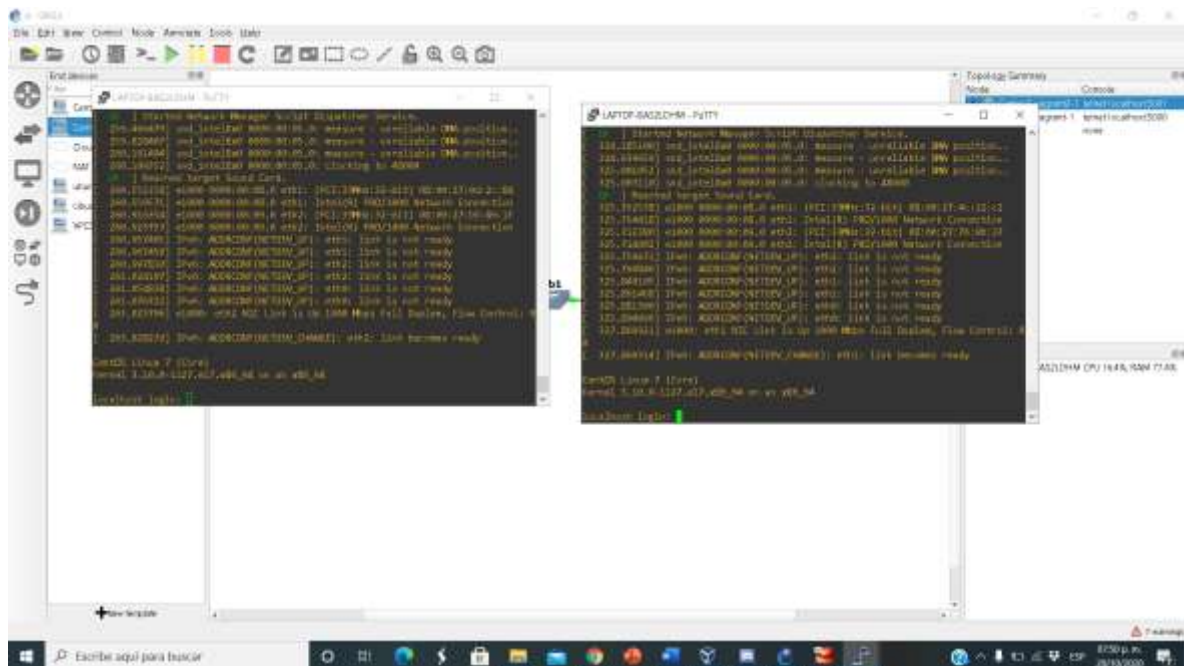
Centos7-vagrant2 [Corriendo] - Oracle VM VirtualBox

Archivo Máquina Ver Entrada Dispositivos Ayuda

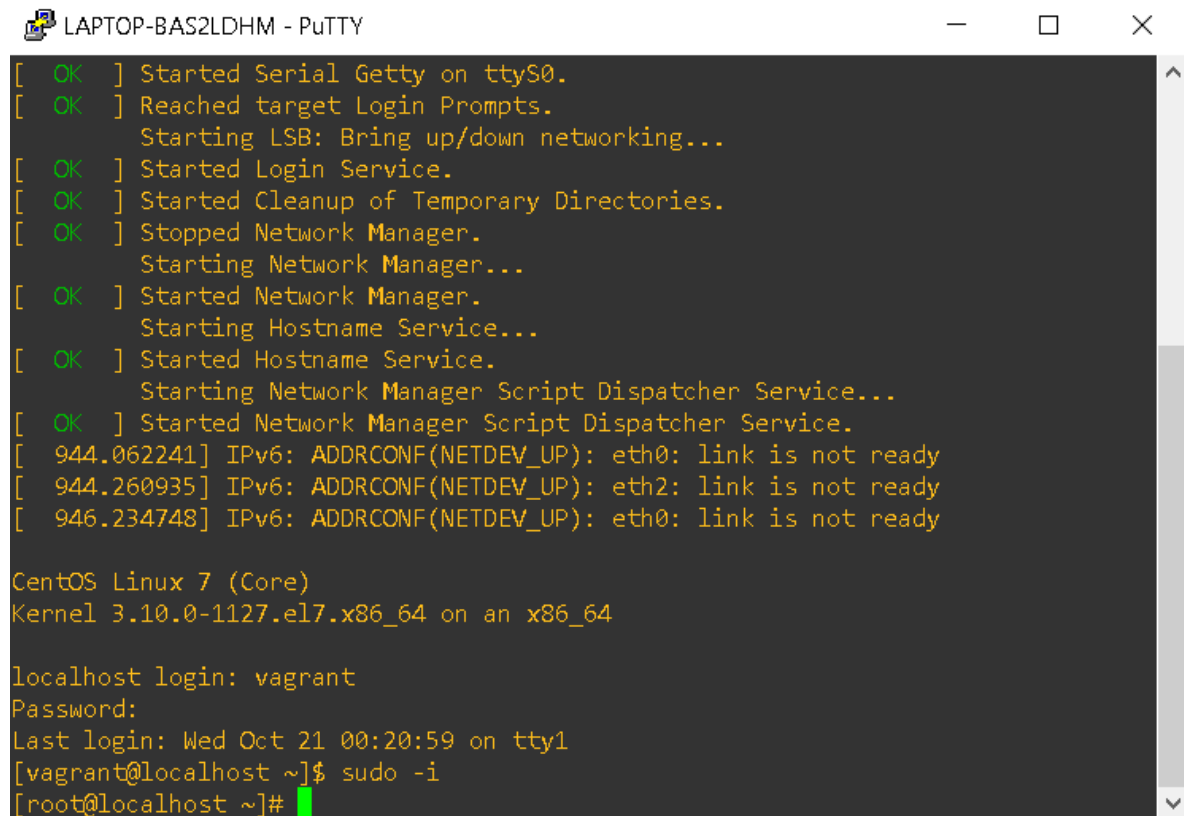
```
CentOS Linux 7 (Core)
Kernel 3.10.0-1127.el7.x86_64 on an x86_64

localhost login: vagrant
Password:
[vagrant@localhost ~]$ yum -y install python2 net-tools
Loaded plugins: fastestmirror
You need to be root to perform this command.
[vagrant@localhost ~]$ yum -y install python2 net-tools
Loaded plugins: fastestmirror
You need to be root to perform this command.
[vagrant@localhost ~]$ sudo -i
[root@localhost ~]# yum -y install python2 net-tools
Loaded plugins: fastestmirror
Determining fastest mirrors
 * base: distro.ibiblio.org
 * extras: mirrors.mit.edu
 * updates: linux-mirrors.fnal.gov
base                                     | 3.6 kB    00:00
extras                                 | 2.9 kB    00:00
updates                                | 2.9 kB    00:00
(1/4): base/7/x86_64/group_gz          | 153 kB    00:00
(2/4): extras/7/x86_64/primary_db      | 206 kB    00:01
(4/4): updates/7/x86_64/pr 51% [=====] 1 1.0 MB/s | 5.6 MB    00:05 ETA
```

CTRL DERECHA



**Fase3** - Usar los scripts de python para conectar las dos VMs usando sockets.



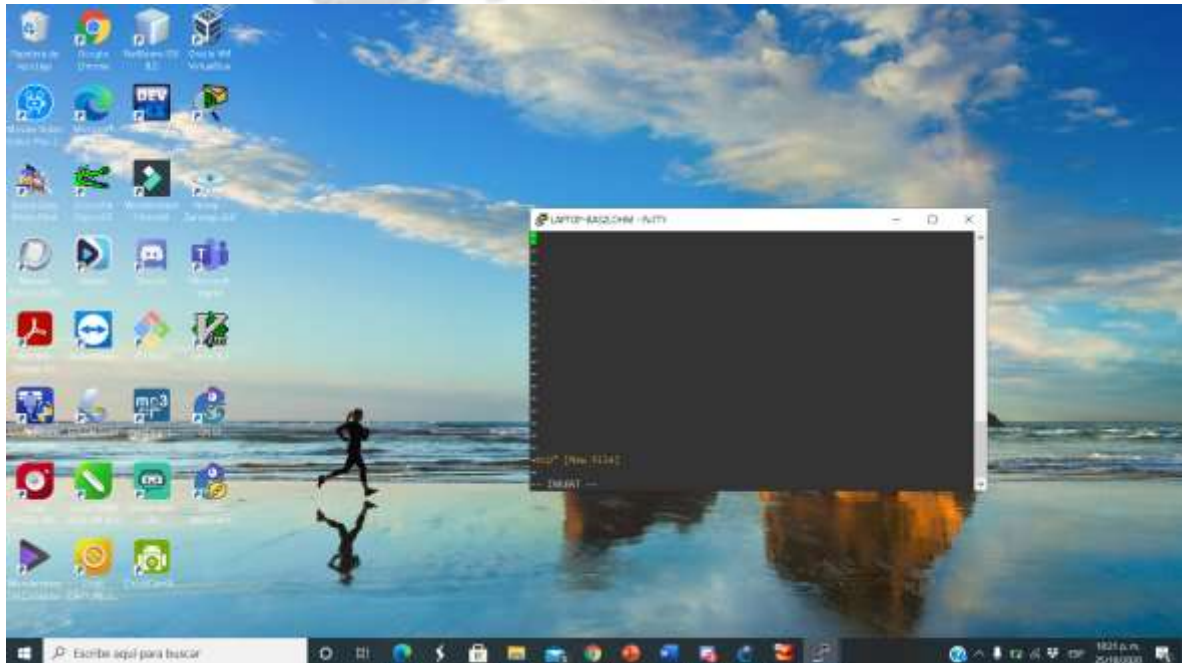
```
LAPTOP-BAS2LDHM - PuTTY

[ OK ] Started Serial Getty on ttyS0.
[ OK ] Reached target Login Prompts.
        Starting LSB: Bring up/down networking...
[ OK ] Started Login Service.
[ OK ] Started Cleanup of Temporary Directories.
[ OK ] Stopped Network Manager.
        Starting Network Manager...
[ OK ] Started Network Manager.
        Starting Hostname Service...
[ OK ] Started Hostname Service.
        Starting Network Manager Script Dispatcher Service...
[ OK ] Started Network Manager Script Dispatcher Service.
[ 944.062241] IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
[ 944.260935] IPv6: ADDRCONF(NETDEV_UP): eth2: link is not ready
[ 946.234748] IPv6: ADDRCONF(NETDEV_UP): eth1: link is not ready

CentOS Linux 7 (Core)
Kernel 3.10.0-1127.el7.x86_64 on an x86_64

localhost login: vagrant
Password:
Last login: Wed Oct 21 00:20:59 on tty1
[vagrant@localhost ~]$ sudo -i
[root@localhost ~]#
```







```

import socket

target_host = "192.168.60.102"
target_port = 2020

# create a socket object
client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

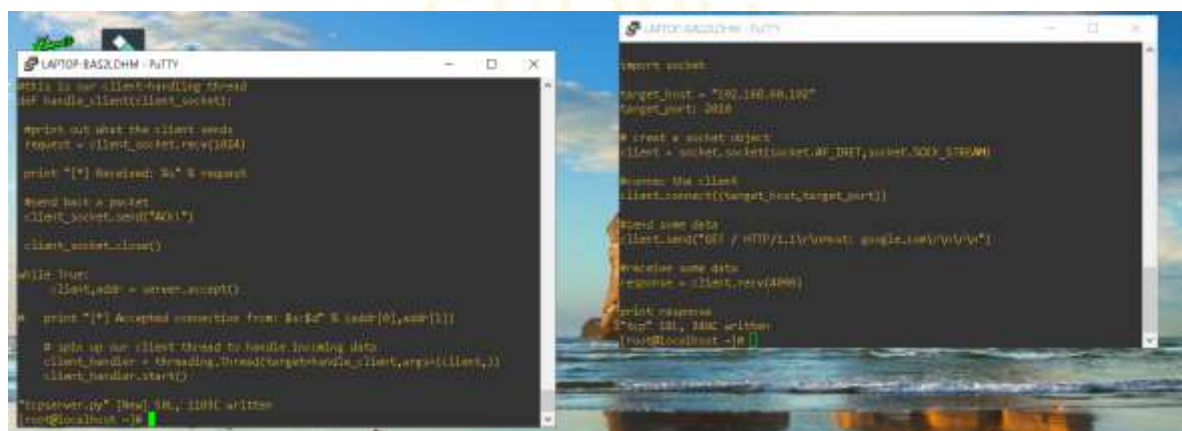
# connect the client
client.connect((target_host, target_port))

# send some data
client.send("GET / HTTP/1.1\r\nHost: google.com\r\n\r\n")

# receive some data
response = client.recv(4096)

print response
"tcp" [New] 18L, 349C written
E173: 1 more file to edit
Press ENTER or type command to continue

```



```

LAPTOP-BASZCHM - PuTTY
# this is our client-handling thread
def handle_client(client_socket):

    # print out what the client sends
    request = client_socket.recv(1024)

    print "[*] Received: %s" % request

    # send back a packet
    client_socket.send("ACK!")

    client_socket.close()

# if __name__ == '__main__':
#     # listen,addr = server.listen()

#     # print "[*] Accepted connection from %s:%s" % (addr[0],addr[1])

#     # sets up our client thread to handle incoming data
#     client_handler = threading.Thread(target=handle_client,args=(client,))
#     client_handler.start()

# # server.py" [New] 18L, 119C written
# root@localhost ~#

```

```

LAPTOP-BASZCHM - PuTTY
import socket

target_host = "192.168.60.102"
target_port = 2020

# create a socket object
client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# connect the client
client.connect((target_host, target_port))

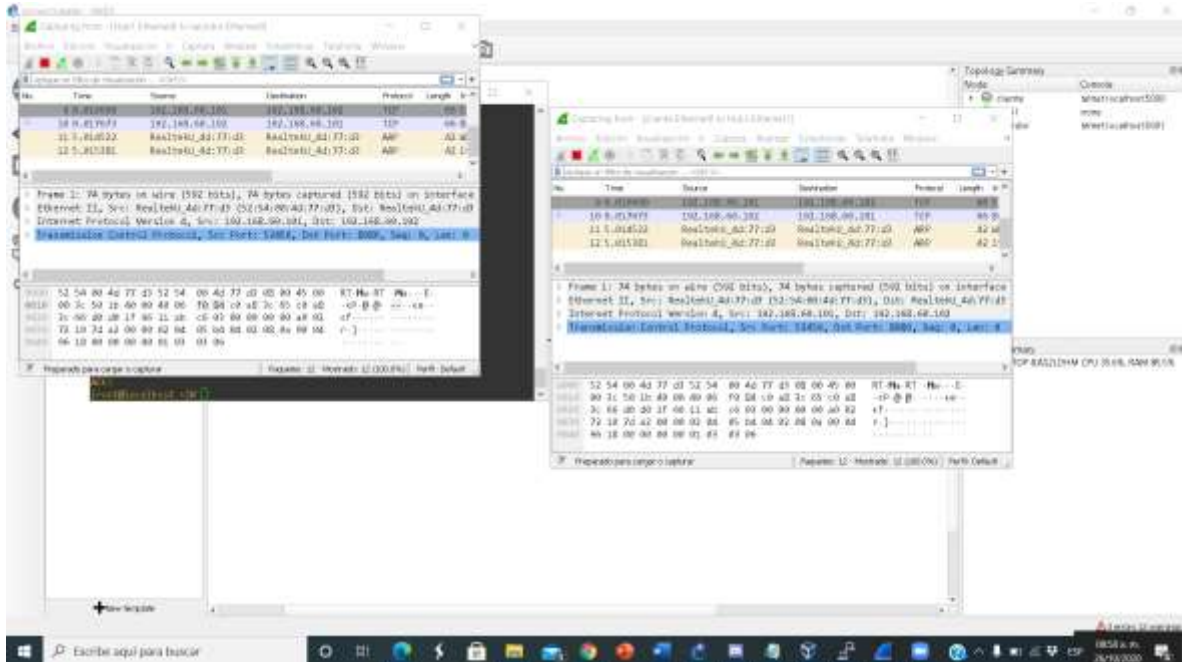
# send some data
client.send("GET / HTTP/1.1\r\nHost: google.com\r\n\r\n")

# receive some data
response = client.recv(4096)

print response
"tcp" [New] 18L, 349C written
root@localhost ~#

```

**Fase4** - Capturar el tráfico de la comunicación entre las dos VMs al momento de utilizar los scripts.



A continuación, podemos ver ya el proyecto ya final terminado usando WireShark para poder ver el trafico de las maquinas virtuales y la comunicación entre ellas.

**Fase 5** - Hacer reporte de conclusiones.

Como lo podemos notar este es un ejemplo sencillo de las comunicaciones que hay hoy en día por ejemplo cuando dos personas están comunicándose a través por medio de dispositivos, en este caso hay dos maquinas los cuales son el **Emisor** y el **Receptor** entre ellos empieza a ver una comunicación, por lo cual pueden mandar los **Mensajes** uno con el otro, también podemos notar en este último proceso el **Medio de comunicación** entre el Cliente y Servidor, viendo los **Protocolos** las reglas a utilizar para que pueda ver esta comunicación gracias a los **Códigos** por medio de sockets para que pueda ver esta prueba.

### ***Conclusión Final***

En este proyecto pudimos ver el cómo instalar un sistema operativo en este caso CentOS 7 sin tener que hacerlo desde VirtualBox, ya que al poner los comando para que ejecute el sistema operativo automáticamente esta corriendo y empezando la instalación, con el Gns3 sirve para poder simular el sistema de comunicación de las maquinas ya instaladas a través de vagrant, poniendo dos computadoras virtuales y un ethernet en este caso un hub para poder realizar este proyecto , usando el programa Putty para poder trabajar con las maquinas virtuales para poder ejecutar los scripts, habilitar los eth0 y agregar las ip a las máquinas, y ya al tener todo eso poder ver el trafico de información o la comunicación entre ellos.



ESCOLTA Y BANDA DE  
GUERRA

Ministerio Tecnologías de Guerra

1999 - 2000 - 2001 - 2002